Задача: разминочная

Название команды: ИЛК

Мартьянов Александр

Архитектор интеграции

Антонова Полина

Архитектор данных

Цветков Алексей

Архитектор системы

Иманаев Алексей

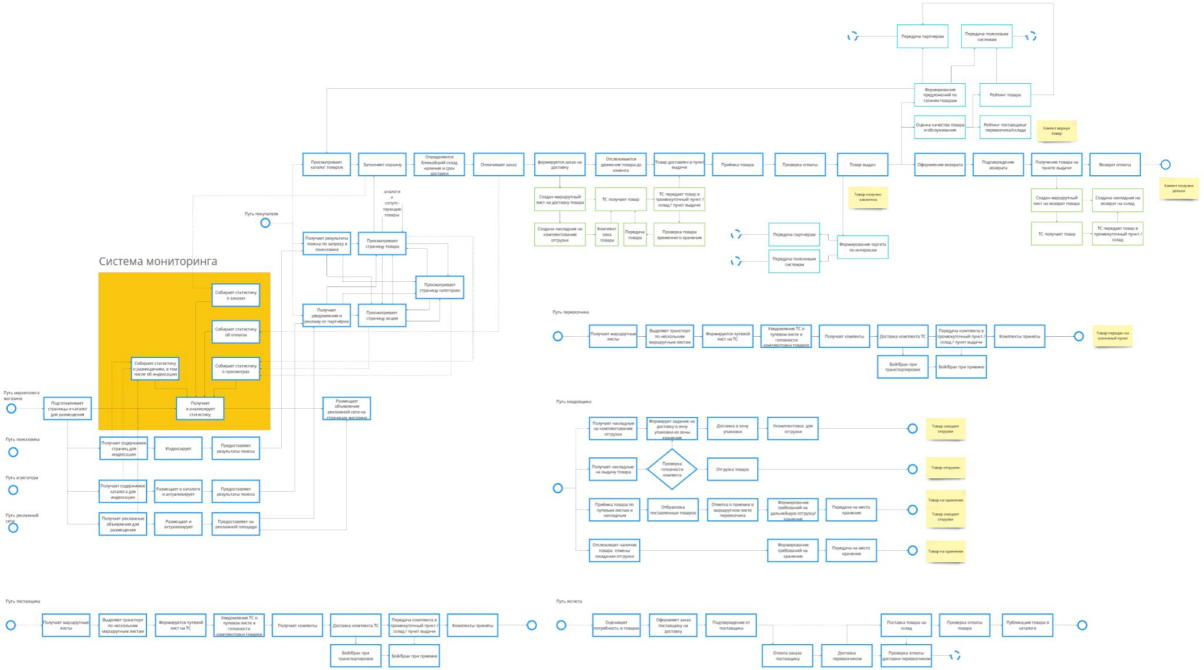
Бизнес-архитектор

Обухов Сергей

Корпоративный архитектор

Решение : функциональная структура

Обширный мозговой штурм позволил составить карту ценности всех функций интернет-магазина.



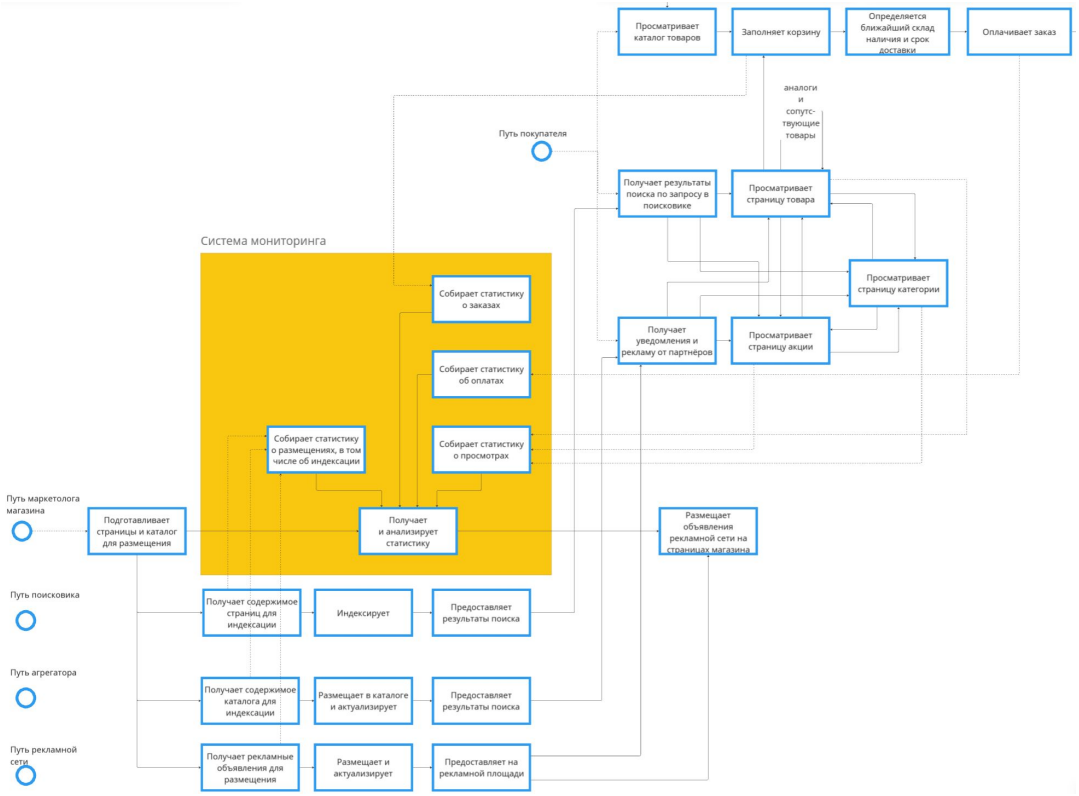
Решение : функциональная структура

Рассмотрим часть, которая относится к маркетинговым функциям и мониторингу.

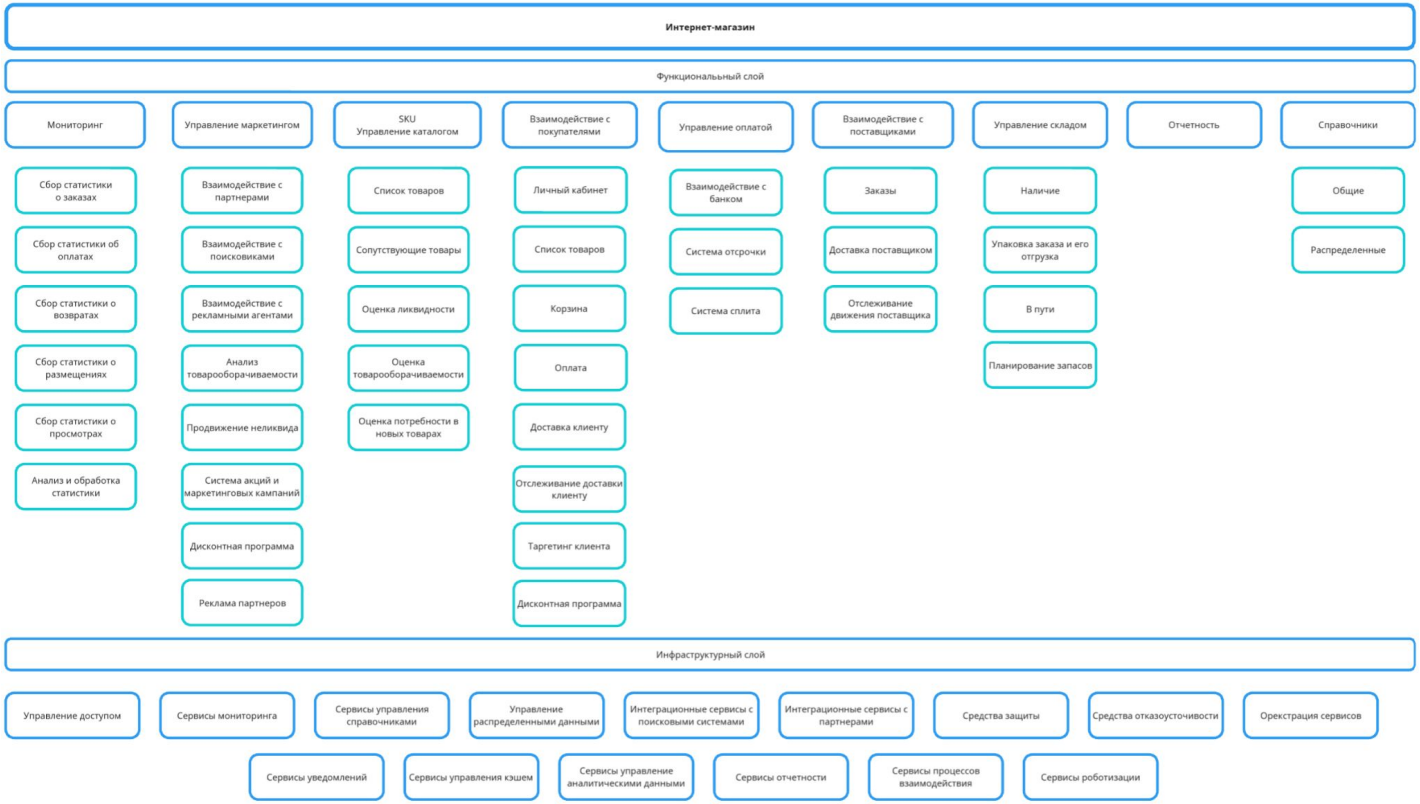
Предусмотрены как мониторинг действий покупателя на сайте, так и функции для управления размещением ассортиментного контента у партнёров: поисковиков, рекламных сетей, агрегаторов.

Эффективность размещения мониторится и анализируется.

В перспективе размещение контента партнёров на страницах магазина с целью монетизации.



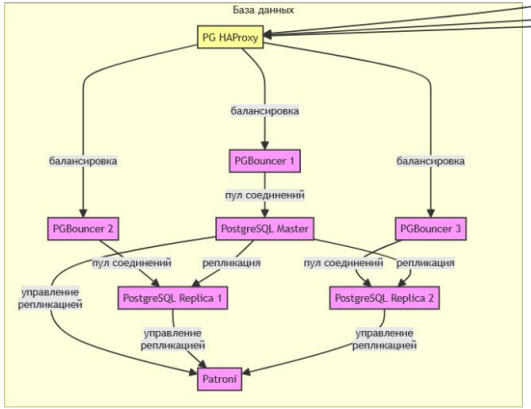
Решение : функциональная архитектура

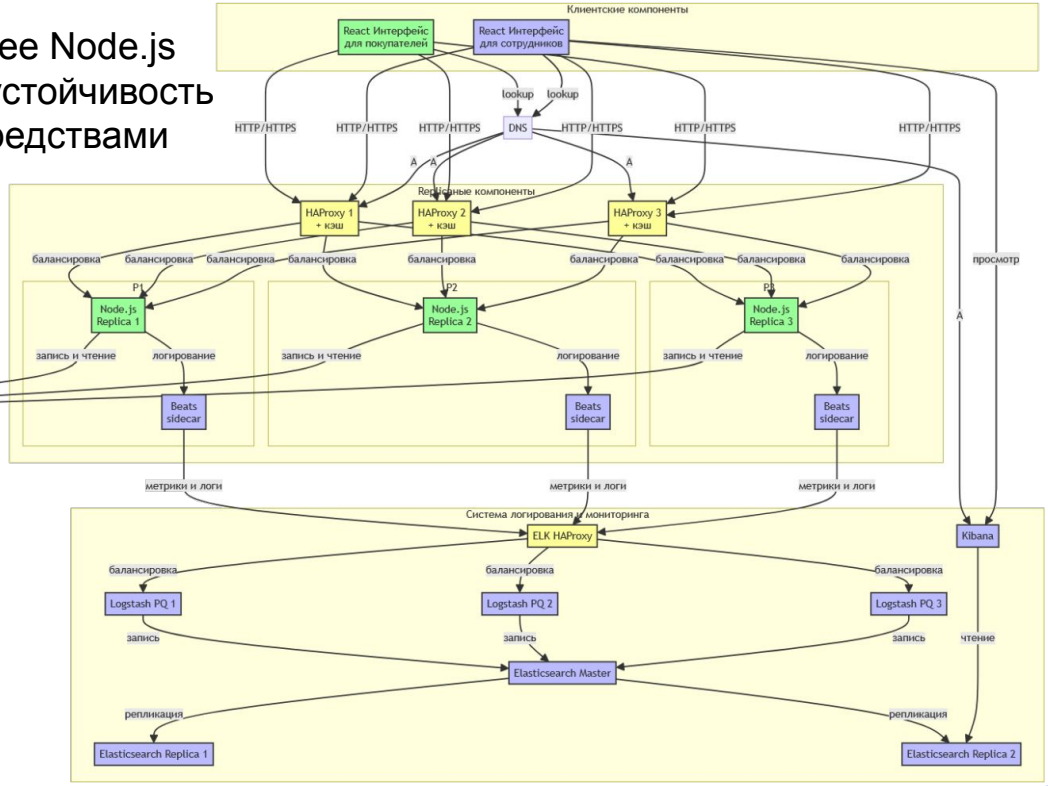


Решение : прикладная архитектура

Решение опирается на существующее Node.js приложение и обеспечивает отказоустойчивость и возможности масштабирования средствами репликации и балансировки.

Мониторинг и аналитика требуют push-модели и богатой модели атрибутов, доступных в ELK стеке.





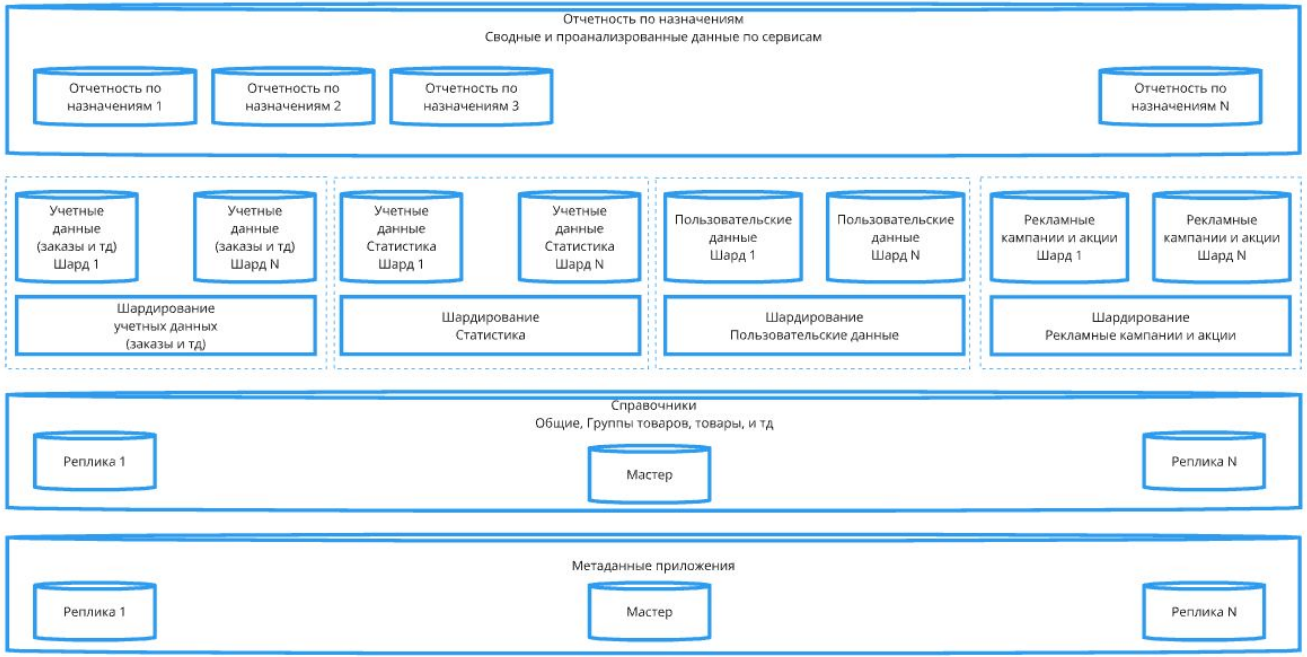
Решение : архитектура данных

Принципы:

- общие и системные сведения через репликацию

- массивные данные через шардирование, для управляющих сервисов применяются сводные БД отчетности по назначениям

- инфраструктурные сервисы приложения обеспечивают целостность, доступность, консистентность данных



Решение : описание API

API пользовательской части интернет-магазина делится на API покупателя и API сотрудника. Эти API уже реализованы в существующем Node.js приложении, например, в виде REST API.

Для данного решения актуально описать API:

- балансировка управляется стандартными API на уровнях DNS и HTTP (HAProxy);
- кэширование управляется стандартным API HTTP (HAProxy);
- отказоустойчивость и масштабирование хранения в PostgreSQL управляются стандартными конфигурациями Patroni, PGBouncer и HAProxy (TCP);
- отказоустойчивость и масштабирование хранения статистики обращений управляются стандартными конфигурациями Logstash Persistent Queues и HAProxy (TCP), а также кластерной конфигурацией ElasticSearch;
- визуализация статистики обращений управляется стандартной конфигурацией и API Kibana, отказоустойчивость Kibana не обеспечивается.

Решение : архитектура развёртывания

Развёртывание системы производится средствами Docker Swarm с перспективой перехода на K8S.

Выделенный контур 0 должен обеспечивать переключение трафика к слоям хранения и статистики посещений в зависимости от ролей экземпляров в механизме репликации.

Контуры 1-3 могут быть развёрнуты и дублированы независимо, а также внутри контура часть компонентов может быть не активна, например, реплика БД или ELK.

Такое деление позволяет начать трансформацию существующего Node.js приложения в схему контура 1, а уже затем в целевую, а также использовать только один контур в среде разработки или на стенде фича-тестирования.